# **Microservices and framework**

**Background:**

**2.1** **Software Application Framework**

Johnson defines a framework as a skeleton of an application that can be modified by an application developer [Johnson97]. With a framework, developers expect to avoid duplicate code, write clean and easily adaptable code, can be scalable code, and be able to focus on writing code specific to their application. A framework can also be classified according to its extensibility; it can be used as a white box or a black box [Schmid96]. Instantiation in white-box frameworks, also known as architecture-driven frameworks, is only feasible through creating new classes. Inheritance or composition can be used to incorporate these classes and codes into the framework [OOP2001]. Black-box frameworks produce instances using configuration scripts. An instantiation automation tool builds the classes and source codes after configuration. It is feasible, for example, to use a graphical wizard to guide the user through the framework instantiation process step by step [OOP2001]. The black box approach does not require framework users to learn details of the framework internals. Figure 1 conceptually illustrates white box and black box frameworks.

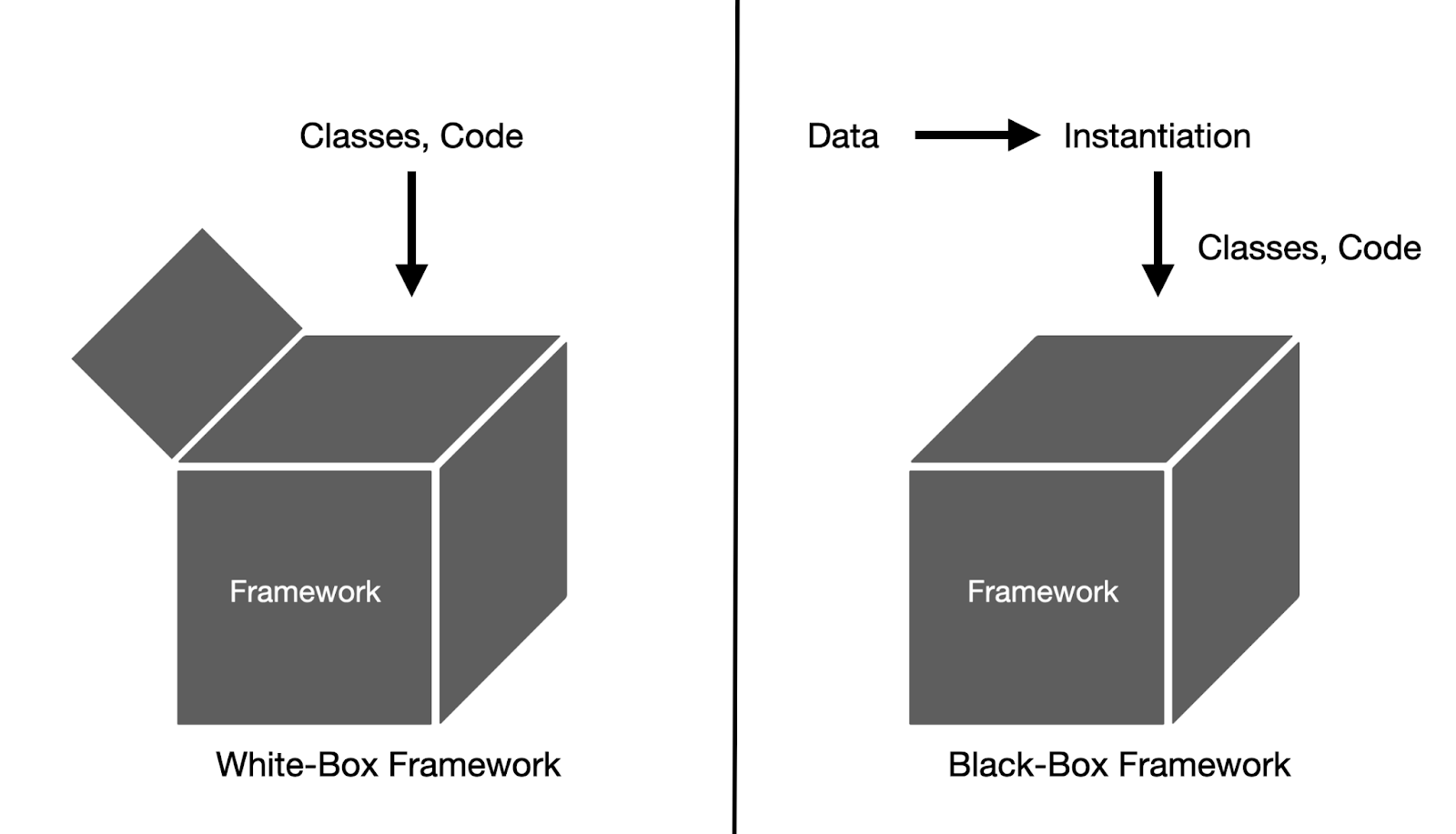


Figure. 1 White-Box Framework and Black-Box Framework

**2.2 Microservices**

In 2014, Martin Fowler [Fowler2014] used the term "Microservices" to define a strategy for developing software applications using a collection of small services. Each service has its own process for serving a specific function, and it interacts with other services via Application Programming Interfaces (APIs) [Villamizar2015]. Unlike monolithic architectures, which are single-code based and one-time deployments, MicroServices have three distinct features. Figure 2 is conceptually described monolithic and microservice architecture.

1. **Independent deployment**

The microservice architecture consists of a collection of independent modules. Unlike the monolith which wraps all operations in a single service, each module in microservices encapsulates its domain logic and contributes to the overall functionality of a system [Hasselbring2017]. The independent deployment improves the flexibility of the development and increases the comprehensibility of the micro service-oriented application.

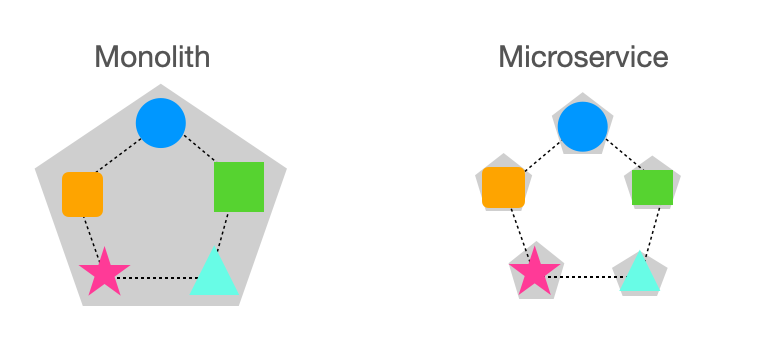


Figure. 2 Monolith and Microservice

1. **Communication mechanism**

The microservices design focuses on lightweight communication mechanisms. Microservices use HTTP request-response with service APIs and lightweight messaging to provide "dumb pipes". Microservices are distinguished from service-oriented architecture (SOA) by this feature [Martin14]. For microservice communication, we describe two different styles of the protocol below.

Synchronous Protocol: The client sends a request and waits for a response from a server. A prominent example of a synchronous protocol is HTTP/HTTPS, in which the client can only continue its function if the server responds [Nish2021].

Asynchronous Protocol: The client code or message sender does not wait for a response from a server. The AMQP protocol, for example, sends a message to a RabbitMQ or to any other messaging broker without waiting for a response [Nish2021].

1. **Language-Independence**

Every programming language has its own set of capabilities and characteristics. Microservices can implement each service in a different language to create an application [Martin14]. It gives developers the option of selecting the best tools.

**2.3 Cloud Computing & Amazon Elastic Compute Cloud**

Cloud computing refers to the on-demand availability of computer system resources, particularly data storage and computational power, without the user having to manage them directly [Mukherjee2019]. Functions in large clouds are frequently dispersed over numerous locations, each of which is a data centre. Cloud computing is available in both public and private versions. Public cloud providers offer their services through the Internet. Private cloud services, on the other hand, cater to a limited number of customers. These services are a network system that provides hosted services [Mathew2014]. A hybrid option is also available, which includes components of both public and private services.

**2.4 What are the significant reimbursements of micro services?**

When it involves a microservice structure there are numerous advantages to be located whilst evaluating it to the vintage monolithic structure? These advantages are the following:

1. **Scaling:**

One of the large flaws with a monolithic device is that it does now no longer scale thoroughly because even the smallest adjustments require a brand new deployment of the complete device. With microservices, handiest the real provider that desires to be scaled calls for a brand new example to be deployed.

1. **Easier to keep:**

When a monolithic gadget receives bigger, it turns extra complicated therefore more difficult to keep as compared to a gadget with microservices wherein every provider is stored minor and particular.

1. **Flexibility:**

As every provider in a microservice structure machine is administered and works independently, everyone may be modified without it affecting the others, permitting sluggish updating or upgrading to take place.

**2.5 Difficulties through micro services:**

The following are the main difficulties through microservices:

* In a gadget constructed with a microservice architecture, the deployed microservices are probably jogging numerous times of an unmarried provider. These offerings may come upon disasters or improvements and because the microservices are dynamically changing, their cope with locations, in addition to the variety of times, may change. This makes it tough to hold songs and offer the best provider that presently is in suitable fitness to the end-user.
* As communique among diverse components of the machine now could be treated with the aid of using sending community calls as opposed to characteristic calls inside the machine, it familiarises a brand new protection risk. Communication can now be greater without problems intercepted and both examine or maybe manipulated if security features aren't implemented.
* When the device is split into a couple of services, functionalities such as classification now emerge as extra complicated to handle. In a microservice device with loads of services, you won't need to recover logs from each unmarried package physically.

**2.6 Selecting frameworks**

Currently, at the market, there are numerous frameworks at exceptional improvement ranges and each framework differs to fill a selected want inside the market. Depending on what kind of utility is being developed, there may be numerous sorts of functionalities that may be labeled both as useful and may be necessary.

First, the one-of-a-kind frameworks had been diagnosed and explored. This becomes performed via way of means of searching through 3 one-of-a-kind types of an internet pages.

* Various network pages concerning microservices, dispensed systems, and service-orientated structure frameworks.
* Google examines with mixtures of the phrases microservice, service-orientated structure, dispensed device frameworks.
* Various academic websites and blogs for developing microservices.

When all of the functionalities were identified, they have been categorised into 4 companies for you to create a matrix. These companies have been:

* Requirements for the microservices that the machine wished whilst transferred from monolithic structure to microservice structure
* Functionalities that might assist countering the cross-reducing worries of microservices.
* Highly prioritized functionalities that have been wanted, in this example via way of means of Ida In front, for the machine that become to be applied as microservices.
* Lower prioritized functionalities.